

AD-A121 730

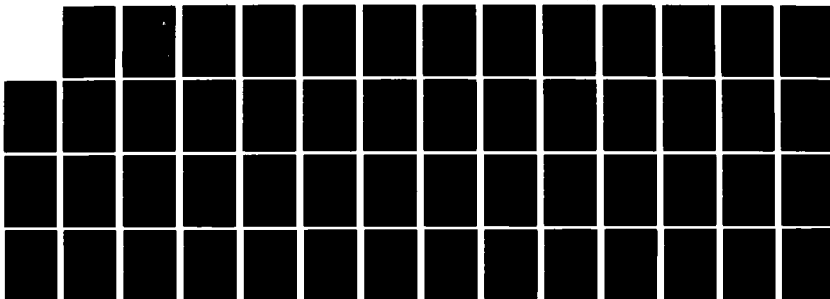
THE RANDOMIZATION TECHNIQUE AS A MODELING TOOL AND  
SOLUTION PROCEDURE FOR. (U) GEORGE WASHINGTON UNIV  
WASHINGTON DC INST FOR MANAGEMENT SCIE..  
D GROSS ET AL. 10 AUG 82 SERIAL-T-467

1/1

UNCLASSIFIED

F/G 12/1.

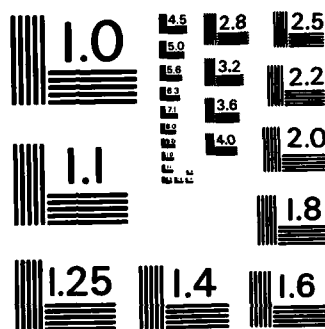
NL



END

FORMED

BY  
DTIC



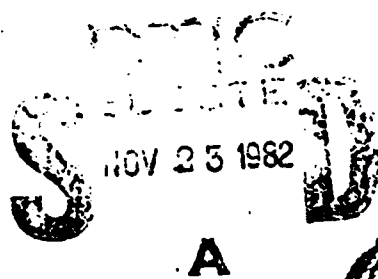
MICROCOPY RESOLUTION TEST CHART  
NATIONAL BUREAU OF STANDARDS-1963-A

AD A121730

(12)



STUDENTS FACULTY STUDY R  
ESEARCH DEVELOPMENT FUT  
URE CAREER CREATIVITY CC  
MMUNITY LEADERSHIP TECH  
NOLOGY FRONTIER DESIGN  
ENGINEERING APPREHENSIVE  
GEORGE WASHINGTON UNIV



82 11 23 048

THE RANDOMIZATION TECHNIQUE AS A MODELING TOOL AND  
SOLUTION PROCEDURE FOR TRANSIENT MARKOV PROCESSES

by

Donald Gross\*†  
Douglas R. Miller\*§

Serial-T-467 ✓  
10 August 1982

The George Washington University  
School of Engineering and Applied Science  
Institute for Management Science and Engineering

Research Supported By

\*Air Force Logistics Management Center  
Contract SCEEE/AFLMC/80-6

†Office of Naval Research  
Program in Logistics  
Contract N00014-75-C-0729, Project NR 347 020

§National Aeronautics and Space Administration  
Grant NAG-1-179

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER T-467	2. GOVT ACCESSION NO. AD-A121730	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) THE RANDOMIZATION TECHNIQUE AS A MODELING TOOL AND SOLUTION PROCEDURE FOR TRANSIENT MARKOV PROCESSES		5. TYPE OF REPORT & PERIOD COVERED SCIENTIFIC
7. AUTHOR(s) DONALD GROSS DOUGLAS R. MILLER		6. PERFORMING ORG. REPORT NUMBER T-467
9. PERFORMING ORGANIZATION NAME AND ADDRESS THE GEORGE WASHINGTON UNIVERSITY INSTITUTE FOR MANAGEMENT SCIENCE AND ENGINEERING WASHINGTON, DC 20052		8. CONTRACT OR GRANT NUMBER(s) SCEEE/AFLMC 80-6 N00014-75-C-0729 NASA NAG-1-179
11. CONTROLLING OFFICE NAME AND ADDRESS AIR FORCE LOGISTICS MANAGEMENT CENTER GUNTER AIR FORCE STATION, AL 36114		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		12. REPORT DATE 10 AUGUST 1982
		13. NUMBER OF PAGES 49
		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report)  APPROVED FOR PUBLIC SALE AND RELEASE; DISTRIBUTION UNLIMITED		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number)  TRANSIENT MARKOV PROCESSES RANDOMIZATION TECHNIQUE		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) The randomization procedure for computing transient solutions to Markov processes (discrete state space, continuous time) is presented. This procedure computes transient state probabilities. It is based on a construction relating a continuous time Markov process to a discrete time Markov chain. Modifications and extensions of the randomization method allow for computation of distributions of first passage and sojourn times in Markov processes, and also computation of expected cumulative occupancy times and expected number (continued)		

DD FORM 1 JAN 73 1473

EDITION OF 1 NOV 65 IS OBSOLETE  
S/N 0102-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

of events occurring during a time interval.

Finite course Markovian queuing systems (machine repair models) are presented as an example; an efficient algorithmic approach is given with some computational results. Additional examples presented are (i) an  $M/E_k/c/K$  queue, and (ii) the reliability of a coherent system with repairable components.

Accession  
NTIS  
1970-1  
1970-1

Doc. No.

Cont.

A

THE GEORGE WASHINGTON UNIVERSITY  
School of Engineering and Applied Science  
Institute for Management Science and Engineering

Abstract  
of  
Serial T-467  
10 August 1982

THE RANDOMIZATION TECHNIQUE AS A MODELING TOOL AND  
SOLUTION PROCEDURE FOR TRANSIENT MARKOV PROCESSES

by

Donald Gross\*†  
Douglas R. Miller\*§

The randomization procedure for computing transient solutions to Markov processes (discrete state space, continuous time) is presented. This procedure computes transient state probabilities. It is based on a construction relating a continuous time Markov process to a discrete time Markov chain. Modifications and extensions of the randomization method allow for computation of distributions of first passage and sojourn times in Markov processes, and also computation of expected cumulative occupancy times and expected number of events occurring during a time interval.

Several implementations of the randomization procedure are discussed. The implementation for a general class of Markov processes which can be described in terms of state space ( $S$ ), event set ( $E$ ), rate vectors ( $R$ ), and target vectors ( $T$ ) --abbreviated as *SERT*-- is presented. This general approach can handle systems whose state spaces are quite large and have sparse generators.

Finite course Markovian queuing systems (machine repair models) are presented as an example; an efficient algorithmic approach is given with some computational results. Additional examples presented are (i) an  $M/E_k/c/K$  queue, and (ii) the reliability of a coherent system with repairable components.

Research Jointly Supported By  
\*Air Force Logistics Management Center  
†Office of Naval Research  
§National Aeronautics and Space Administration

THE GEORGE WASHINGTON UNIVERSITY  
School of Engineering and Applied Science  
Institute for Management Science and Engineering

THE RANDOMIZATION TECHNIQUE AS A MODELING TOOL AND  
SOLUTION PROCEDURE FOR TRANSIENT MARKOV PROCESSES

by

Donald Gross  
Douglas R. Miller

0. INTRODUCTION

Markov processes (with discrete state space over continuous time) are good models for many stochastic systems, including certain queuing systems, inventory systems, reliability and maintenance models, etc. The analyses of these Markov process models are often restricted to steady-state behavior, that is, systems in equilibrium. However, there are many instances in which the transient behavior of the process is important:

- (i) Transient behavior is often encountered because of changes in operating conditions of the system due to exogenous environmental conditions or internal control of the system, such as the opening or closing of a queuing system.
- (ii) In some systems with time homogeneous stochastic behavior, the convergence to steady state is so slow that the equilibrium behavior is not indicative of system behavior.

- (iii) For systems in steady state, there are some transients of interest, such as busy periods for queuing systems.
- (iv) To obtain equilibrium results for some models, transient behavior is useful. For example, the steady-state behavior of regenerative processes is characterized by the behavior during an individual cycle (transient behavior); see Ross [1970, Theorem 5.8].

The primary transient quantities of interest are the time-dependent state probabilities. From these, time-dependent moments can be calculated, etc. (Other quantities of interest are distributions of first passage times, expected cumulative occupancy times during a time interval, and expected number of a certain class of events occurring during a time interval.) In general, the transient state probabilities of a Markov process can be computed by solving a system of linear first order differential equations. Nice analytical solutions rarely exist. Even for the M/M/1 queue, the solution comes out as a rather formidable expression in terms of modified Bessel functions; see Gross and Harris [1974]. In general, numerical methods must be used to solve for the transient state probabilities.

This paper focuses on the approach called "randomization," first introduced as a computational method by Grassmann [1977]. It is a general method for computing transient probabilities of Markov processes with finite state spaces, for example, finite queues (finite source or finite capacity). The method can be used on many infinite state Markov processes by approximating the process with a finite state Markov process; for example, an M/M/c/ $\infty$  queue can be approximated by an M/M/c/K queue with

sufficiently large  $K$ . The randomization approach has the advantage of having a probabilistic interpretation that can be exploited in efficient model generation and computations for a broad class of Markov models.

The paper is organized as follows. Section 1 presents transient theory for Markov processes and reviews some of the solution approaches. Section 2 presents the idea of subordinating a Markov chain to a Poisson process, the basis of the randomization approach. Section 3 develops these ideas for birth-death processes. Section 4 gives details encountered in randomization computational algorithms. Section 5 presents a general modeling approach called *SERT* which can be used with the randomization approach; three examples are given. Section 6 gives formulae and methodology for computing other transient quantities in addition to state probabilities. Section 7 gives a randomization implementation for sparse systems. Section 8 contains some concluding remarks.

## 1. TRANSIENT STATE PROBABILITIES OF MARKOV PROCESSES

Let  $\{X(t), t \geq 0\}$  be a Markov process on a finite state space  $S = \{0, 1, \dots, N\}$ . The stochastic evolution of this process is characterized by its generator,

$$Q = \begin{pmatrix} -q_0 & q_{01} & q_{02} & \cdots & q_{0N} \\ q_{10} & -q_1 & q_{12} & \cdots & q_{1N} \\ \vdots & & & & \\ q_{N0} & q_{N1} & q_{N2} & & -q_N \end{pmatrix},$$

where the rows sum to zero, i.e.,

$$\sum_{j \neq i} q_{ij} - q_i = 0$$

and the element  $q_{ij}$  is

$$q_{ij} = \lim_{\Delta t \rightarrow 0} p_{ij}(\Delta t) / \Delta t ,$$

where

$$p_{ij}(\Delta t) = \Pr\{X(t + \Delta t) = j \mid X(t) = i\} .$$

These Markov processes include realistic classes of Markovian queuing models such as finite waiting room (capacity) queues, finite source (machine repair) queues, cyclic queues, and closed queuing networks. (Also, infinite Q matrices can be approximated by truncation for some large value of N for many models.) For example, the Q matrix for a general birth-death system with finite capacity is given by

$$Q = \begin{bmatrix} -\lambda_0 & \lambda_0 & 0 & 0 & 0 & \dots & \dots & 0 \\ \mu_1 & (-\lambda_1 - \mu_1) & \lambda_1 & 0 & 0 & \dots & \vdots \\ 0 & \mu_2 & (-\lambda_2 - \mu_2) & \lambda_2 & 0 & \dots & \vdots \\ & & & \vdots & & & \vdots \\ & & & & & & 0 \\ & & & & & & \vdots \\ \vdots & & & & 0 & \dots & \mu_{N-1} & (-\lambda_{N-1} - \mu_{N-1}) & \lambda_{N-1} \\ 0 & \dots & & & \dots & & \mu_N & -\mu_N \end{bmatrix} \quad (1)$$

Let  $\pi_j(t) = \Pr\{X(t) = j\}$  and  $\underline{\pi}(t) = (\pi_0(t), \pi_1(t), \dots, \pi_N(t))$ ; then the Kolmogorov forward equations (see Karlin and Taylor [1975, p. 152],

or Heyman and Sobel [1982, p. 295], for example) are

$$\pi'(t) = \pi(t)Q. \quad (2)$$

For a finite state space  $S = \{0, 1, \dots, N\}$  we have a system of  $N + 1$  first order, linear differential equations to solve.

Given  $\pi(0)$ , there are several approaches to the solution of (2). Classical numerical analysis techniques for the numerical integration of (2) include Runge-Kutta, or predictor-corrector methods, etc., as discussed by Arsham, Balana, and Gross [1981], Grassmann [1977], or Maron [1982]. These methods have the disadvantage that they are formal numerical analysis techniques and ignore any exploitable probabilistic structure of the problem (for example, the structure of  $Q$ ).

Another approach is to note that

$$\pi(t) = \pi(0) e^{Qt}$$

is the solution of (2), where by definition

$$e^{Qt} = I + \sum_{n=1}^{\infty} \frac{Q^n t^n}{n!}.$$

A common method of computing  $e^{Qt}$  is based on the spectral representation of  $Q$  (see Çinlar [1975, pp. 367-370], or Karlin and Taylor [1975, pp. 539-541], for example). This involves computation of all the eigenvalues and eigenvectors of  $Q$ , a very tricky feat when the state space  $S$  is large. Liou [1966] proposed an iterative computational procedure for computing the terms of

$$\pi(t) = \sum_{n=0}^{\infty} \pi(0) \frac{Q^n t^n}{n!} \quad (3)$$

recursively, but as Grassmann [1977] points out, the existence of

negative diagonal elements in  $Q$  leads to possibly severe round off errors. Moler and Van Loan [1978] give a survey of methods for computing the exponential of a matrix. They are primarily concerned with matrices that are small enough to be stored in core memory (the randomization method can handle much larger matrices for sparse systems). They also point out how difficult it is to solve this computational problem in general.

The randomization method is especially tailored for solution of (2) when  $Q$  is the generator of a Markov process and  $\pi$  is a probability vector. It has a probabilistic interpretation that can be used to bound the truncation error arising from the infinite series in (3) and which allows the algorithm to work only with positive numbers (thus minimizing round off errors). Furthermore, the probabilistic meaning is an aid in setting up the model and executing the algorithm for a large class of systems. The randomization approach is based on the subordination of a Markov chain to a Poisson process.

## 2. MARKOV CHAINS SUBORDINATED TO POISSON PROCESSES

Let  $\{Y_n, n = 0, 1, 2, \dots\}$  be a Markov chain on a countable state space  $S$  with transition matrix  $\tilde{P}$ . Let  $\{N(t), t \geq 0\}$  be the counting process of a Poisson process with rate  $\Lambda$ , i.e.,  $N(t)$  equals the number of occurrences in  $[0, t]$  and  $\Pr\{N(t) = n\} = e^{-\Lambda t} (\Lambda t)^n / n!$ . Assume that  $\{Y_n\}$  and  $\{N(\cdot)\}$  are independent. Define the process  $\{X(t) = Y_{N(t)}, t \geq 0\}$ . It is well known that  $\{X(t), t \geq 0\}$  is a Markov process (continuous time) on  $S$  with generator  $Q = \Lambda(\tilde{P} - I)$  and the same initial distribution. This construction is referred to as a *Markov chain subordinated to a Poisson process*; see Feller [1971, p. 345], or Çinlar [1975, p. 236].

Cohen [1969] refers to these processes as *derived Markov chains*. It is also occasionally referred to as a *Markov chain on an underlying Poisson process*. Feller [1971] refers to the events of the Poisson process as a "randomized operational time" and the construction as "randomization by Poisson distributions;" hence the name *randomization* is applied to this construction as well as to the computational technique that exploits it.

Conversely, let  $\{X(t), t \geq 0\}$  be a Markov process on a countable state space  $S$  with generator  $Q$ . Furthermore, assume that  $X(\cdot)$  is *uniformizable*, i.e., the diagonal elements of  $Q$  are uniformly bounded, and let  $\Lambda = \sup_{i \in S} q_i$ . Then there exists a Markov chain (discrete time)  $\{Y_n, n = 0, 1, \dots\}$  on  $S$  with transition matrix  $\tilde{P} = Q/\Lambda + I$  and a Poisson process  $\{N(t), t \geq 0\}$  with rate  $\Lambda$ , which are independent of each other, such that the process  $\{Y_{N(t)}, t \geq 0\}$  and  $\{X(t), t \geq 0\}$  have the same finite dimensional distributions and are thus probabilistically identical. (See Heyman and Sobel [1982, pp. 310-311], for example.)

The above construction gives a very useful computational formula for the transient probabilities of a uniformizable Markov process. Conditioning over the number of occurrences of the Poisson process in  $[0, t]$  and using the law of total probability gives

$$\begin{aligned} \pi_s(t) &= P\{X(t) = s\} \\ &= P\{Y_{N(t)} = s\} \\ &= \sum_{n=0}^{\infty} P\{Y_{N(t)} = s \mid N(t) = n\} P\{N(t) = n\} \\ &= \sum_{n=0}^{\infty} P\{Y_n = s\} \frac{e^{-\Lambda t} (\Lambda t)^n}{n!}. \end{aligned}$$

Defining  $\phi_s(n) = P\{Y_n = s\}$  and  $\tilde{\phi}(n) = (\phi_1(n), \phi_2(n), \dots)$ , this equation becomes

$$\tilde{\pi}(t) = \sum_{n=0}^{\infty} \tilde{\phi}(n) \frac{e^{-\Lambda t} (\Lambda t)^n}{n!}. \quad (4)$$

Recalling that  $\tilde{\pi}(0) = \tilde{\phi}(0)$  and  $\tilde{P}$  is the transition matrix of  $\{Y_n, n = 0, 1, \dots\}$ , equation (4) can be written as

$$\tilde{\pi}(t) = \sum_{n=0}^{\infty} \tilde{\pi}(0) \tilde{P}^n \frac{e^{-\Lambda t} (\Lambda t)^n}{n!}. \quad (5)$$

This formula appears in Cinlar [1975, p. 259], and Cohen [1969, pp. 46-47]. It is apparently originally due to Jensen [1953].

This construct relating a Markov process to a randomized Markov chain, originally due to Jensen [1953], is useful in a least two other general areas of applied probability. It is useful for developing stochastic inequalities between processes and monotonicity properties of processes (see Keilson [1979], Keilson and Kester [1977], Kirsten [1977], and Sonderman [1980], for example). This construction is also useful for establishing equivalences between discrete and continuous time Markov decision processes (see Howard [1960], Lippman [1975], Serfozo [1979], and Veinott [1969]).

### 3. A DIRECT PROBABILISTIC DEVELOPMENT OF THE RANDOMIZATION CONSTRUCT FOR A BIRTH-DEATH PROCESS

As an example to illustrate this construction, let us consider the birth and death process whose generator  $Q$  is given in (1). We have a birth-death process, which means that the system state (the population

size) can change by +1 or -1; that is, if a birth occurs, the size increases by one, while if a death occurs, the size decreases by one. In a small instant of time, if the system is in state  $s$ , the probability of an increase to  $s + 1$  is  $\lambda_s \Delta t + o(\Delta t)$ , and the probability of a decrease to  $s - 1$  is  $\mu_s \Delta t + o(\Delta t)$ . By the definition of conditional probability, letting  $X(t)$  represent the random variable "size of the population at time  $t$ ," we have

$$\begin{aligned} & \Pr\{X(t + \Delta t) = s + 1 \mid X(t) = s \text{ and a transition occurs in } (t + \Delta t)\} \\ &= \frac{[\lambda_s \Delta t + o(\Delta t)]\pi_s(t)}{[\lambda_s \Delta t + \mu_s \Delta t + o(\Delta t)]\pi_s(t)} \\ &= \frac{\lambda_s \Delta t + o(\Delta t)}{(\lambda_s + \mu_s)\Delta t + o(\Delta t)} . \end{aligned}$$

Taking the limit as  $\Delta t$  goes to zero tells us that given that a transition occurs when the system is in state  $s$ , the probability that it will be a birth (increase to  $s + 1$ ) is  $\lambda_s / (\lambda_s + \mu_s)$ , where  $\mu_0 \equiv 0$  and  $\lambda_N \equiv 0$ . Thus the probability that, given a transition occurs when the system is in state  $s$ , it is a death (decrease to  $s - 1$ ) is  $\mu_s / (\lambda_s + \mu_s)$ .

The holding time in state  $s$  for this Markov process is the minimum of two exponential random variables, the interarrival time with parameter  $\lambda_s$  and the service time with parameter  $\mu_s$ ; hence the holding time is also an exponential random variable with parameter  $\lambda_s + \mu_s$ .

Thus we can view the system as a Markov process with exponential holding times [mean of  $(\lambda_s + \mu_s)^{-1}$  for holding in state  $s$ ] and transition probabilities of  $\lambda_s / (\lambda_s + \mu_s)$  and  $\mu_s / (\lambda_s + \mu_s)$  for going "up" one or "down" one, respectively, from state  $s$ . Furthermore, it can be shown

that the occurrence of "birth" or "death" is independent of the holding time.

Suppose we wish to generate this process, for example, to simulate it by Monte Carlo means. We could first generate the transition time occurrence and then generate, using simple Bernoulli probabilities, the change of state--that is, whether the process increases or decreases its state by one. The generation of the transition times is somewhat complicated in that we would have to sample from an exponential distribution with a state-dependent parameter  $\lambda_s + \mu_s$ .

To avoid this, we can reproduce this process in the following way. Consider the minimum mean holding time (time until the next transition occurs) of the process. This will correspond to the maximum value of  $\lambda_s + \mu_s$ , or equivalently, the minimum diagonal element of  $Q$ , the generator of the process. Call this value  $\Lambda$ . Let us denote the diagonal elements of the  $Q$  matrix by  $-q_s$ , that is,

$$q_s = \begin{cases} \lambda_0 & (s = 0) \\ \lambda_s + \mu_s & (s = 1, 2, \dots, N-1) \\ \mu_N & (s = N) \end{cases}$$

To reproduce our desired transition occurrence process, we could generate a true Poisson process with rate  $\Lambda$  (exponential holding times with constant mean  $1/\Lambda$ ) and then "thin" the process to get the desired state-dependent transition rate. By thinning, we mean that whenever an occurrence is generated by the Poisson ( $\Lambda$ ) process, if we are in state  $s$  we draw from a Bernoulli probability distribution with success probability  $q_s/\Lambda$ , where a success indicates counting the occurrence of a transition

and a failure (probability  $1 - q_s/\Lambda$ ) indicates ignoring the occurrence. This thinning procedure on the Poisson ( $\Lambda$ ) process generates our desired underlying state-dependent transition process.

Thus, to reproduce our process we can

- i. generate a Poisson process with rate  $\Lambda$ ;
- ii. thin the Poisson ( $\Lambda$ ) process by a Bernoulli "switch" with acceptance probability  $q_s/\Lambda$  and rejection probability  $1 - q_s/\Lambda$ ;
- iii. generate the state change (up or down one unit) by another Bernoulli switch with an up probability of  $\lambda_s/(\lambda_s + \mu_s)$  and a down probability of  $\mu_s/(\lambda_s + \mu_s)$ .

Denote the Poisson process as  $\{N(t), t \geq 0\}$ , i.e.,  $N(t)$  equals the number of occurrences in  $[0, t]$ , and let  $Y_n$  equal the state of the system after the  $n$ th occurrence of the Poisson process, i.e.,  $Y_n$  equals  $X(T_n)$ , where  $T_n = \min(t : N(t) \geq n)$ . Viewing the process in this manner will allow us to derive the randomization computing algorithm that will yield  $\pi(t)$ .

Consider an element of the transient state probability vector  $\pi(t)$ , say  $\pi_s(t) = P\{X(t) = s\}$ . Letting  $p_{is}(t) = \Pr\{X(t) = s \mid X(0) = i\}$  gives

$$\pi_s(t) = \sum_{i \in S} \pi_i(0) p_{is}(t). \quad (6)$$

Now considering our process as described above, we have, using the law of total probability,

$$\begin{aligned}
 p_{is}(t) &= \sum_{n=0}^{\infty} \Pr\{Y_n = s \mid Y_0 = i\} \Pr\{N(t) = n\} \\
 &= \sum_{n=0}^{\infty} \tilde{p}_{is}^{(n)} \frac{e^{-\Lambda t} (\Lambda t)^n}{n!}
 \end{aligned} \tag{7}$$

where  $\tilde{p}_{is}^{(n)}$  represents the probability of going from state  $i$  to state  $s$  in  $n$  occurrences of the Poisson process. But

$$\begin{aligned}
 \tilde{p}_{is}^{(1)} &= \Pr\{\text{of going from } i \text{ to } s \text{ in one occurrence}\} \\
 &= \Pr\{\text{of accepting the occurrence as a transition}\} \\
 &\times \Pr\{\text{of transiting from } i \text{ to } s \mid \text{a transition takes place}\} \\
 &= \begin{cases} \frac{q_i}{\Lambda} \cdot \frac{\lambda_i}{\lambda_i + \mu_i} = \frac{\lambda_i}{\Lambda} & (s = i + 1) \\ \frac{q_i}{\Lambda} \cdot \frac{\mu_i}{\lambda_i + \mu_i} = \frac{\mu_i}{\Lambda} & (s = i - 1) \\ 1 - \frac{\lambda_i + \mu_i}{\Lambda} & (s = i) \\ 0 & (\text{elsewhere}) \end{cases}
 \end{aligned}$$

Denote the matrix with elements  $\tilde{p}_{is}^{(1)}$  by  $\tilde{P}$ ; note that  $\tilde{P} = Q/\Lambda + I$ . We can obtain  $\tilde{p}_{is}^{(n)}$  as elements of a matrix formed by multiplying  $\tilde{P}$  by itself  $n$  times, i.e.,

$$\{\tilde{p}_{is}^{(n)}\} = [\tilde{P}]^n.$$

Substituting this into equations (6) and (7) yields

$$\tilde{\pi}(t) = \sum_{n=0}^{\infty} \tilde{\pi}(0) \tilde{P}^n \frac{e^{-\Lambda t} (\Lambda t)^n}{n!}$$

which is equation (5). Thus we have illustrated the general theory with this special case of a birth-death process.

#### 4. ALGORITHMIC IMPLEMENTATION OF THE RANDOMIZATION FORMULA

The randomization algorithm is based upon equation (4):

$$\pi(t) = \sum_{n=0}^{\infty} \phi(n) \frac{e^{-\Lambda t} (\Lambda t)^n}{n!} . \quad (4)$$

There are several considerations that arise in implementing this technique: (i) truncation of the infinite series, (ii) efficient calculation of  $\phi_n$ , (iii) multiple time points, (iv) changes in  $Q$ , (v) an approach to systems with small state spaces, (vi) an approach to systems with infinite state spaces, and (vii) efficient computation of moments and probabilities of subsets.

##### 4.1 Truncation of the Randomization Formula

Equation (4) involves an infinite series. Thus, we must obviously truncate the series at some point, say  $T$ , so that the computational formula becomes

$$\pi(t) \approx \sum_{n=0}^T \phi(n) \frac{e^{-\Lambda t} (\Lambda t)^n}{n!} . \quad (8)$$

We can set  $T$  to bound the error due to truncation (see Arsham, *et al.* [1981], for example) by satisfying

$$1 - e^{-\Lambda t} \sum_{n=0}^T \frac{(\Lambda t)^n}{n!} \leq \epsilon ,$$

where  $\epsilon$  is the desired error control. This would guarantee  $\pi_s(t)$  and all other probabilities to be accurate to within  $\epsilon$  ( $\epsilon = 10^{-3}$ , for example, guarantees at least two decimal place accuracy).

#### 4.2 Efficient Calculation of $\phi(n)$

To use equation (8),  $\phi(n)$  must be computed. Recall that  $\phi_s(n) = P\{Y_n = s\}$ , where  $\{Y_n, n = 0, 1, \dots\}$  is the discrete time Markov chain. Thus the randomization computational technique really reduces to computation of the transient probabilities  $\phi$  of the Markov chain.

Computation of the  $\phi(n)$ 's is naturally a recursive procedure, since

$$\phi(n+1) = \phi(n)\tilde{P}, \quad (9)$$

a fact that follows from elementary Markov chain theory. In general, the calculation in equation (9) is a straightforward matrix multiplication. However, most specific systems have sparse  $\tilde{P}$  matrices and thus efficient computation of equation (9) should exploit the sparseness of  $\tilde{P}$ . Grassmann [1977,1982] uses sparse matrix techniques, although he does not explain them in detail. Melamed and Yadin [1981] give a recursion for efficiently computing  $\phi(n)$  for Jacksonian queuing networks which exploits the sparseness of  $Q$ . We give a general modeling approach called SERT in Section 5 of this paper--it is a method applicable to many sparse systems. In Section 7 we give a general approach that completely exploits the sparseness.

There is one other feature of the efficient calculation of equation (9) that should be mentioned. In many systems, it is most natural to use a multidimensional description of the state space. However, equation (9) can be calculated in less time if the state space is "linearized," i.e., the states are ordered into a one-dimensional state space. This efficiency is gained because less arithmetic is involved in locating and storing values in a vector than in an array. Also, since these

multidimensional state spaces often have irregular shapes, some storage space is wasted by the dummy states required in filling the state space out as a rectangular array.

#### 4.3 Transients at Multiple Time Points

In a typical application we will want to compute transient quantities at a sequence of time points  $t_1 < t_2 < \dots < t_{m-1} < t_m < t_{m+1} < t_\ell$  instead of just at a single time point. Typically, these time points will be on a lattice, i.e.,  $t_{m+1} - t_m = h$  for all  $m$ . Without loss of generality, we can label our time scale so that  $h = 1$ . Consequently, we would like to compute  $\pi(1), \pi(2), \dots, \pi(\ell)$  and related measures over the time periods.

Using the randomization algorithm  $\ell$  times, once for each  $\pi(m)$ , is very inefficient because the  $\phi$ 's will be recomputed each time. Thus, the efficient approach is to compute the  $\phi$ 's recursively and as each is computed, add a term to each of the  $\ell$  series

$$\sum_{n=0}^i \phi(n) \frac{e^{-\Lambda t_m} (\Lambda t_m)^n}{n!}, \quad m = 1, 2, \dots, \ell,$$

until the desired truncation points  $T(\epsilon, t_m)$ ,  $m = 1, 2, \dots, \ell$ , are reached.

#### 4.4 Changes in the Generator Q

A phenomenon that may occur in the study of transient behavior is time-nonhomogeneity of the stochastic process. For a Markov process, one manifestation of this phenomenon is the change of the generator at some discrete point in time: on  $[0, \ell)$ , the process may have generator  $Q$  and then on  $[\ell, \ell']$  have generator  $Q'$ . In order to compute the transient

probabilities in this case, we compute  $\pi(1), \dots, \pi(\ell)$  as before and then use  $\pi(\ell)$  as the initial probability vector for a second problem involving a process with generator  $Q'$ .

Thus, letting  $\tilde{P}' = Q'/\Lambda' + I$  be the transition matrix of a Markov chain  $\{Y'_n, n = 0, 1, 2, \dots\}$ , which has initial distribution  $\phi'(0) = \pi(\ell)$  and marginals  $\phi'_s(n) = \Pr\{Y'_n = s\}$ , we have

$$\phi'(n+1) = \phi'(n)\tilde{P}'$$

and

$$\pi(\ell+k) = \sum_{n=0}^{T'} \phi'(n) \frac{e^{-\Lambda'k} (\Lambda'k)^n}{n!}, \quad k = 1, 2, \dots$$

When applying the randomization algorithm to Markov processes whose generator changes at discrete times during the time interval of interest,  $[0, \ell]$ , there is a slight difficulty in dealing with the truncation error  $\varepsilon$ . It is desired that all probabilities computed have a maximum error equal to the inputted value of  $\varepsilon$ ; this is achieved by working over each subinterval with epsilons which sum to an inputted  $\varepsilon$  and are proportional to the appropriate time subinterval. For example, suppose we are interested in transient behavior over  $[0, 15]$  and the generator changes at times 6 and 10. Then the problem will be solved in three parts: first on  $[0, 6]$ , epsilon equal to  $(6/15)\varepsilon$  is used; then on  $[6, 10]$ , epsilon equal to  $(4/15)\varepsilon$  is used; and finally on  $[10, 15]$ , epsilon equal to  $(5/15)\varepsilon$  is used. This approach will guarantee that all probabilities related to the interval  $[0, 15]$  will have a computational error of  $\varepsilon$  or less.

#### 4.5 Systems with Small State Spaces

For systems with small state spaces which do not have a high degree of sparseness, there is another way to compute transient probabilities on a lattice of time points,  $t_m = m$ ,  $m = 0, 1, \dots, \ell$ . In this case we modified the above algorithm to compute the matrix of probabilities  $P(1) = \{p_{is}(1)\}$ , where

$$p_{i,s}(1) = P\{X(1) = s \mid X(0) = i\}$$

and using previous notation we have

$$P(1) = \sum_{n=0}^{\infty} \tilde{P}^n \frac{e^{-\Lambda} \Lambda^n}{n!}$$

giving us a computational formula after the appropriate truncation of the infinite sum.

Assuming the process is time-homogeneous over  $[0, \ell)$ , i.e., the generator  $Q$  does not change, we use the initial probability vector  $\tilde{\pi}(0)$  and then recursive computation to obtain

$$\tilde{\pi}(1) = \tilde{\pi}(0)P(1)$$

$$\tilde{\pi}(2) = \tilde{\pi}(1)P(1)$$

$$\vdots$$

$$\tilde{\pi}(\ell) = \tilde{\pi}(\ell - 1)P(1) .$$

The program SASTRANX described in Balana, *et al.* [1982] computes transient probabilities for a machine repair system that experiences changes in rates at a finite number of discrete time points in this way.

#### 4.6 Systems with Infinite State Spaces

Some systems, such as infinite capacity queuing systems, will have infinite state spaces. Melamed and Yadin [1981] consider this type of system. In this case it is necessary to truncate the state space. This will introduce an approximation error into the calculations. This error can be bounded by collapsing all the truncated states into one absorbing state and then applying the randomization algorithm. The probability that the system is in the absorbing state will be a bound on the error introduced by truncating the state space. For an example, see Section 5.2

#### 4.7 Efficient Computation of Moments and Subset Probabilities

In the transient analysis of a Markov process, if the only performance measures of interest are moments or probabilities of certain subsets of the state space, then it is unnecessary to compute the entire state probability vector  $\pi(t)$ .

Suppose, for example, we are only interested in  $P\{X(t) \in A\}$  for some subset  $A$  of the state space. It will be more efficient to forego computation of  $\pi(t)$ 's and to compute probabilities at occurrence times of the underlying Poisson process

$$A_n = \sum_{s \in A} \phi_s(n) = P\{X_n \in A\} = P\{X(T_n) \in A\}$$

and then use

$$A(t) = P\{X(t) \in A\} = \sum_{n=0}^{T(\epsilon, t)} A_n P\{N(t) = n\}.$$

Similarly, transient measures of the form

$$E[f(X(t_m))] \quad m = 1, 2, \dots, \ell$$

can be more efficiently computed by foregoing  $\pi(t_m)$ ,  $m = 1, \dots, \ell$  and instead using

$$E[f(Y_n)] = \sum_{s \in S} f(s) \phi_s(n)$$

and

$$E[f(X(t))] = \sum_{n=0}^{T(\epsilon, t)} E[f(Y_n)] P(N(t) = n) .$$

An example of this type of performance measure is the expected number of machines operating at time  $t$  in a machine repair model.

## 5. SERT MODELING PROCESS AND RANDOMIZATION

We now present our approach to modeling continuous parameter Markov processes, which utilizes the randomization procedure in an efficient manner and enables us to determine the transient quantities of interest discussed in the previous sections. We call the approach SERT, since it involves the four concepts,

$S$  = state space

$E$  = set of types of events

$R$  = set of transition rate vectors

(one for each element in  $E$ ; each vector has  $|S|$  components)

$T$  = set of target state vectors

(one for each element in  $E$ ; each vector has  $|S|$  components)

We now describe an algorithm based on the above for a general class of continuous-time Markov processes on discrete state spaces.

Consider a Markov process on a discrete state space  $S$ . It has  $E$  types of events that may occur:  $e_1, e_2, \dots, e_E$ . Let  $\bar{E} = \{e_1, e_2, \dots, e_E\}$ . For each  $e_j \in \bar{E}$  there exist two vectors, the vector of transition rates  $\tilde{r}^j$ , and the vector of target states  $\tilde{t}^j$ ,  $j = 1, 2, \dots, E$ . Thus, when the process is in state  $s \in S$ ,  $r_s^j$ , the  $s$ th component of  $\tilde{r}^j$ , is the rate at which event  $e_j$  will occur and consequently put the system into state  $t_s^j$ .

The characterization of the process in terms of states, events, rates, and targets is a very fruitful way of working with a broad class of Markov processes, both from the point of view of defining or describing the process and for use in algorithmic solution methodologies.

The holding time in state  $s$  is exponentially distributed with rate  $r_s = \sum_{j=1}^E r_s^j$ . Let  $\Lambda = \max_{s \in S} r_s$  be the rate of an underlying Poisson process (this corresponds to the maximum diagonal element of the  $Q$  matrix but with this approach it is not necessary to generate the  $Q$  matrix per se). Define a Markov chain on this Poisson process whose transitions are the result of  $E + 1$  distinct events;  $e_\emptyset$  is the "null event." The transition probabilities corresponding to different events are

$$p_s^j \equiv \Pr\{\text{event of type } j \mid \text{transition (occurrence of Poisson } (\Lambda) \text{ process) when in state } s\} = r_s^j / \Lambda \quad j = 1, 2, \dots, E$$

$$p_s^\emptyset \equiv \Pr\{\text{null event, i.e., remaining in state } s \mid \text{transition when in state } s\} = 1 - r_s / \Lambda$$

and

$$t_s^\emptyset \equiv \text{the state to which system goes when a null event occurs} = s.$$

Transient probabilities of the continuous-time Markov process are computed from the transient probabilities of the Markov chain, using equation (8),

$$\tilde{\pi}(t) = \sum_{n=0}^{T(\epsilon, t)} \tilde{\phi}(n) P(N(t) = n) .$$

The vectors  $\tilde{\phi}(n)$  are computed recursively:

$$(i) \quad \tilde{\phi}(0) = \tilde{\pi}(0)$$

(ii)  $\tilde{\phi}(n+1)$  is computed from  $\tilde{\phi}(n)$  as follows:

$$(a) \quad \phi_s(n+1) = \phi_s(n) \cdot p_s^\phi, \quad s \in S$$

then

(10)

(b) for  $j = 1, 2, \dots, E$ , and for  $s \in S$ , add

$$\phi_s(n) \cdot p_s^j \quad \text{to} \quad \phi_{t_s^j}(n+1) .$$

For a sequence of times  $0 < t_1 < t_2 < \dots < t_\ell$  over which the process is time-homogeneous, compute  $\tilde{\pi}(t_m)$ ,  $m = 1, \dots, \ell$  simultaneously by initializing each to 0 and adding  $\tilde{\phi}(n)P(N(t_m) = n)$  to the  $m$ th vector as the  $\tilde{\phi}(n)$ 's are computed recursively. At a discrete point of time-nonhomogeneity, use the last  $\tilde{\pi}(t_\ell)$  as the initial vector and repeat the process for time points in the next time interval for which time homogeneity exists.

In the following subsection, we illustrate these concepts on three examples, the first dealing with a classical machine repair problem, the second an  $M/E_k/c$  queue, and the last a maintained reliability system.

### 5.1 A Machine Repair Example

Consider the classical machine repair with spares model (see, for example, Gross and Harris [1974]). Since this is a finite birth-death process, the generator is given by equation (1), where

$$\lambda_s = \begin{cases} M\lambda & (0 \leq s \leq Y) \\ (N - s)\lambda & (Y \leq s \leq Y+M \equiv N) \\ 0 & (s \geq N) \end{cases}$$

$$\mu_s = \begin{cases} s\mu & (0 \leq s \leq c) \\ c\mu & (c \leq s \leq N) \end{cases}$$

and

$1/\lambda$  = mean time to failure of a machine

$1/\mu$  = mean time to repair of a machine

$M$  = desired number of machines operating

$Y$  = number of spare machines

$c$  = number of repairmen (service channels).

Let us consider a specific problem for which  $M = 5$ ,  $Y = 3$ ,  $c = 2$ ,  $\lambda = 0.2$ , and  $\mu = 0.4$ . Thus,

$$Q = \begin{bmatrix} -1.0 & 1.0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ .4 & -1.4 & 1.0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & .8 & -1.8 & 1.0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & .8 & -1.8 & 1.0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & .8 & -1.6 & .8 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & .8 & -1.4 & .6 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & .8 & -1.2 & .4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & .8 & -1.0 & .2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & .8 & -.8 \end{bmatrix}$$

The machine repair problem has a special structure that can be used to develop an implementation of the randomization algorithm that exploits the sparseness of  $Q$ , which is common to many applications in queuing theory: there is a very small class of distinct events that are responsible for all the state changes of the system, namely, there are only two types of events, "failure of a machine," and "completion of repair of a machine." If the state of the system is described as the number of machines in repair, say  $s$ , then  $s$  is an element of the state space  $S$  ( $s \in S$ ), where  $S = \{0, 1, 2, \dots, 8\}$  in our example, the event "failure" corresponds to transition  $s \rightarrow s + 1$ , the event "repair" corresponds to transition  $s \rightarrow s - 1$ , and the respective transition rates (or intensities) are found on the superdiagonal and the subdiagonal of  $Q$ . Thus instead of describing the stochastic behavior of this system with a  $9 \times 9 = 81$  element  $Q$  matrix, we can describe it in terms of the state space  $S = \{0, 1, \dots, 8\}$ , two events  $E = \{f, r\}$  -- "failure" and "repair" -- and for each event a nine-component vector of transition rates, namely,  $\tilde{r}^f$  and  $\tilde{r}^r$ , and a nine-component vector indicating the resulting (target) state when each event occurs to the system in each possible state, namely,  $\tilde{t}^f$  and  $\tilde{t}^r$ . Thus the vector of transition rates with which a "failure" occurs is

$$\tilde{r}^f = (1.0, 1.0, 1.0, 1.0, .8, .6, .4, .2, 0.0)$$

and the vector of target states given a failure is

$$\tilde{t}^f = (1, 2, 3, 4, 5, 6, 7, 8, 8)$$

(where we use the convention that the target is the same as the original if the transition intensity is 0.0). Similarly, for the "repair" event

the transition rates are

$$\underline{r}^r = (0.0, .4, .8, .8, .8, .8, .8, .8, .8)$$

and the targets are

$$\underline{t}^r = (0, 0, 1, 2, 3, 4, 5, 6, 7) .$$

Thus the behavior of the process can be described using  $4 \times 9 = 36$  numbers instead of  $9 \times 9 = 81$  elements of  $Q$ . This description of the process in terms of the state space, events, transition rate vectors, and target vectors is our *SERT* procedure.

We implement the randomization algorithm using this idea of rate vectors and target vectors for each event, cf. Grassmann's QUE package (Grassmann [1982]). Consider the Markov chain with transition probability matrix  $\tilde{P}$  on the underlying Poisson process: it has three events, "failure," "repair," and "null event" (no transition). We can see that this Markov chain can be described in terms of transition probability vectors for each event and vectors of target states for each event. Note that the vectors of targets are, of course, the same for the original Markov process and the Markov chain on the underlying Poisson process. The vectors of transition probabilities for the three events are:

$$\underline{p}^f = (.556, .556, .556, .556, .444, .333, .222, .111, 0)$$

$$\underline{p}^r = (0, .222, .444, .444, .444, .444, .444, .444, .444)$$

$$\underline{p}^\phi = (.444, .222, 0, 0, .112, .222, .333, .444, .556) .$$

Note that the vectors  $\underline{p}^f$ ,  $\underline{p}^r$ ,  $\underline{p}^\phi$ ,  $\underline{t}^f$ ,  $\underline{t}^r$ , and the value  $\Lambda = 1.8$  completely determine the process. Note that  $\underline{p}^f = \underline{r}^f / \Lambda$  and  $\underline{p}^r = \underline{r}^r / \Lambda$ , where  $\underline{r}^f$  is the superdiagonal and  $\underline{r}^r$  is the subdiagonal of the  $Q$  matrix.

The fundamental relationship for the randomization algorithm to be used is equation (8), namely,

$$\pi(t) = \sum_{i=0}^T \phi(i) P(N(t) = i) .$$

The vectors  $\phi(n)$ ,  $n = 1, 2, \dots$ , are computed recursively;  $\phi_0 = \pi(0)$ , which is assumed given. To compute  $\phi(n+1)$  from  $\phi(n)$ , recall that  $\phi(n+1) = \phi(n)\tilde{P}$ . We want to avoid direct matrix multiplication by  $\tilde{P}$  because it is so sparse. Instead we use the algorithm given as (10) based on the vectors of transition probabilities  $p_s^f$ ,  $p_s^r$ ,  $p_s^\phi$ , and the vectors of target states  $t_s^f$  and  $t_s^r$ .

We restate the algorithm for this example as follows. For this system there are three steps:

1. For each  $s \in S$ , set  $\phi_s(n+1) = \phi_s(n) \cdot p_s^\phi$
2. For each  $s \in S$ , add  $\phi_s(n) \cdot p_s^f$  to  $\phi_{t_s^f}(n+1)$
3. For each  $s \in S$ , add  $\phi_s(n) \cdot p_s^r$  to  $\phi_{t_s^r}(n+1)$

The notation takes a bit of getting used to. Subscripts always refer to an element of a vector. For example,

$$\begin{aligned} \phi_s(n) &= \text{sth element of } \phi(n) \\ &= \text{Pr}\{\text{system state is } s \text{ (s machines in repair) after } n \\ &\quad \text{occurrences of the underlying Poisson } (\Lambda) \text{ process}\} \\ t_s^f &= \text{sth element of the failure target vector, } t^f; \text{ that is,} \\ &\quad \text{the state to which the system transits given it is in} \\ &\quad \text{state } s \text{ and a failure occurs} \end{aligned}$$

$$\begin{aligned}\phi_{t_s}^{f(n+1)} &= t_s^f \text{th element of the } \phi(n+1) \text{ vector} \\ &= \text{Pr}\{\text{system state is } t_s^f \text{ after } n+1 \text{ occurrences of the} \\ &\quad \text{underlying Poisson } (\Lambda) \text{ process}\} .\end{aligned}$$

A little thought reveals that this algorithm is exactly the same as  $\phi(n)\tilde{P}$ , but avoids all the multiplication by zeroes off the three main diagonals, and also avoids storage of an  $N \times N$   $\tilde{P}$  matrix.

Consider the  $\tilde{P}$  matrix for our example. We can calculate  $\phi(n+1) = \phi(n)\tilde{P}$  by straight matrix multiplication. For example,

$$\begin{aligned}\phi_4(n+1) &= 0 \cdot \phi_0(n) + 0 \cdot \phi_1(n) + 0 \cdot \phi_2(n) + .556\phi_3(n) + .112\phi_4(n) \\ &\quad + .444\phi_5(n) + 0 \cdot \phi_6(n) + 0 \cdot \phi_7(n) + 0 \cdot \phi_8(n) \\ &= .556\phi_3(n) + .112\phi_4(n) + .444\phi_5(n) .\end{aligned}$$

Notice all the zero multiplications. The algorithm method builds up the  $\phi(n+1)$  vector as follows:

Algorithm Step	$\phi(n+1)$ Element					
	$\phi_0(n+1)$	$\phi_1(n+1)$	$\phi_2(n+1)$	$\phi_3(n+1)$	$\phi_4(n+1)$	...
1	$.444\phi_0(n)$	$.222\phi_1(n)$	$0 \cdot \phi_2(n)$	$0 \cdot \phi_3(n)$	$.112\phi_4(n)$	...
	+	+	+	+	+	
2		$.556\phi_0(n)$	$.556\phi_1(n)$	$.556\phi_2(n)$	$.556\phi_3(n)$	...
		+	+	+	+	
3	$0 + .222\phi_1(n)$	$.444\phi_2(n)$	$.444\phi_3(n)$	$.444\phi_4(n)$	$.444\phi_5(n)$	...

The column sums directly give  $\phi_s(n+1)$ ; for example,

$$\phi_4(n+1) = .112\phi_4(n) + .556\phi_3(n) + .444\phi_5(n)$$

and hence the six zero multiplications are avoided.

For the machine repair problem, this algorithmic approach has been written in a FORTRAN program called REPTRAN1 (see Gross, *et al.* [1982]). It can handle discrete changes in failure and repair rates. Over an interval for which  $Q$  is constant it simultaneously computes  $\pi(t)$  for a lattice of time points, by initializing the vectors to  $\underline{0}$  and then adding  $\phi(n)P(N(t) = n)$  as the  $\phi(n)$ 's are computed recursively, until the truncation point is reached. The transient probability vectors for the next interval of constant  $Q$  is computed using the probability vector of the last point in the preceding interval as the initial vector for this interval. (It computes  $\Lambda$  from the formula

$$\Lambda = \begin{cases} M\lambda + c\mu & \text{if } c \leq Y \\ M\lambda + Y\mu & \text{if } c > Y, \lambda > \mu \\ (M + Y - c)\lambda + c\mu & \text{if } c > Y, \lambda < \mu \end{cases}$$

instead of searching for the maximum value of  $q_s$ .) It also plots availability (the probability of at least  $M$  machines up = 1 - probability of at most  $Y$  down) versus time. Figure 1 shows the plot for this example, where  $\lambda$  increases by 50% at time 6 and  $\mu$  increases by 50% at time 10.

## 5.2 A Queuing Example: $M/E_k/c$ System

In this section we give a SERT model of an  $M/E_k/c$  queuing system. The transient state probabilities can then be solved using the randomization algorithm given in the previous section.

First consider the state description of the system. One possibility is

$$s = (s_1, s_2, \dots, s_c, q)$$

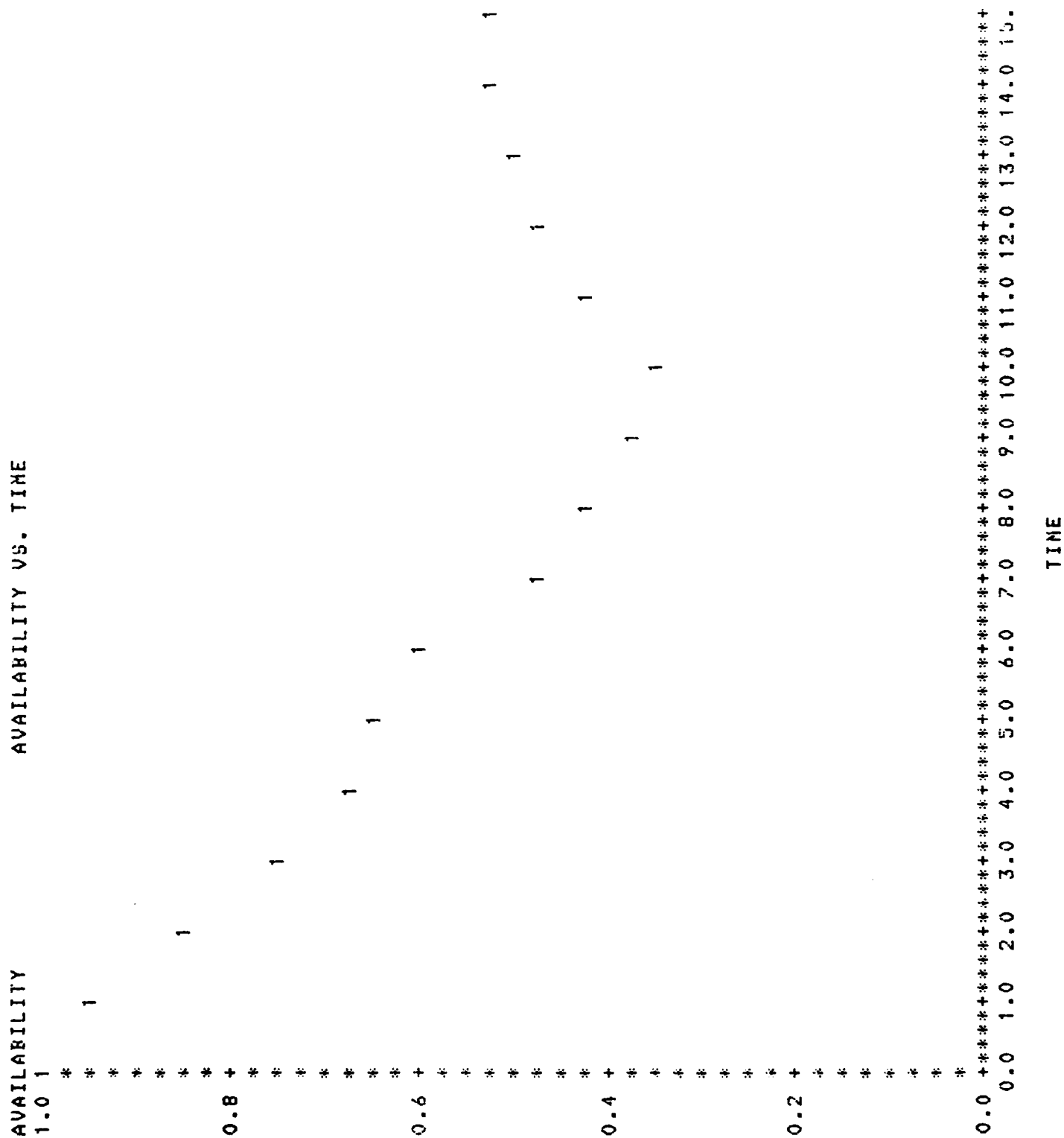


Figure 1.--Availability vs. time for machine repair example with a 50% increase in  $\lambda$  at  $t = 6$  and a 50% increase in  $\mu$  at  $t = 10$ .

where  $s_i$  equals the Erlang stage of the  $i$ th service channel ( $s_i = 0$  signifying that the  $i$ th channel is empty) and  $q$  equals the number of customers waiting in the queue. We can partition the state space into two sets corresponding to whether the queue is empty,  $S_0$ , or whether there is a positive number of customers in the queue,  $S_+$ . For states in  $S_0$ ,  $q$  must be 0 but each  $s_i$  may be any integer between 0 and  $k$ ; thus  $S_0$  consists of  $(k+1)^c$  states, which we assume to be arranged in lexicographical order. Since the queuing system has infinite capacity,  $S_+$  is clearly infinite and thus must be truncated in order to use the randomization algorithm, as discussed in Section 4.6. Let  $S_{+,K}$  correspond to the subset of  $S_+$  for which  $q \leq K - c$ ; thus  $S_0 \cup S_{+,K}$  are the states of an  $M/E_k/c/K$  system. The elements in  $S_{+,K}$  must have  $1 \leq q \leq K - c$  and  $1 \leq s_i \leq k$ , and consequently there are  $(K - c)k^c$  states in  $S_{+,K}$ , which we also assume are arranged in lexicographical order. Finally, we shall add one more state, an absorbing state "a", which will be useful in obtaining a bound on the error due to truncation. Let  $S_a = \{a\}$ . Whenever the system exceeds  $K$  customers, it is absorbed into  $S_a$ . It will be necessary to choose  $K$  by observing the amount of probability that is absorbed by  $S_a$  and adjusting  $K$  to make this equal to or less than some small value such as  $10^{-2}$  or  $10^{-3}$ . The state space will be

$$S = S_K = S_0 \cup S_{+,K} \cup S_a.$$

Now let us consider the set of possible events  $E$ . There are two basic types of events: arrivals and completions of a stage of service. There is the possibility of describing several different systems based upon the mechanism by which a customer selects a service channel when

more than one is idle. We shall model this as a classical overflow problem: the arriving customer attempts to enter channel 1; if channel 1 is busy he tries channel 2, and so forth. If all channels are busy he overflows into the queue. (In our truncated state space, if  $K$  customers are in the system, the arrival forces the system into the absorbing state  $S_a$ .) Thus the state change due to an arrival is deterministic and it suffices to have a single arrival event  $A$ . It is necessary to have a separate event of completion of a stage of service for each service channel, because the target state depends on the channel where the event occurred. Thus we have stage completion events  $C_1, C_2, \dots, C_c$  and the event set has  $c + 1$  elements

$$E = \{A, C_1, C_2, \dots, C_c\}.$$

Now let us consider the collection  $R$  of rate vectors. We have  $c + 1$  rate vectors, one corresponding to each event in  $E$ . In the modeling activity we think in terms of a rate function, i.e., the transition rate is a function of the state of the system. (In Sections 5.2 and 5.3 we thus use functional notation instead of vector notation to describe the rates and targets in the SERT description.) First consider the rate function for the arrival event  $A$ : arrivals keep coming regardless of the state of the system; thus

$$r_A(s) = \lambda \quad s \in S.$$

In our standard numerical implementation of the randomization procedure we would use the above function to generate the rate vector with components  $r_s^A = \lambda$ ,  $s \in S$ , i.e.,

$$\tilde{r}^A = (\lambda, \lambda, \dots, \lambda)$$

which would be used in the calculation. (There is always some flexibility in specifying the SERT description. In this case we could have set  $r_A(a) = 0$  instead of  $\lambda$ , but it will make the programming easier to have the arrival rate equal to  $\lambda$ , independent of the state of the system. A similar consideration holds for the rate of service-stage completion.) Now consider the event  $C_i$ , completion of a stage of service at channel  $i$ . Since the server in channel  $i$  completes stages according to a Poisson process with rate  $\mu_i$ ,

$$r_{C_i}(s) = \mu_i \quad s \in S.$$

(Again it is convenient for programming purposes to have a constant rate. In this case if the server completes a stage of service while the channel is empty, it remains empty, constituting a dummy event.)

Finally, let us consider the collection of all target vectors or functions,  $T$ . Each of these gives the target state as a function of current state for each event in  $E$ . First consider the arrival event. Since we are modeling the system as a classic overflow system

$$\begin{aligned} t_A(s_1, s_2, \dots, s_c, q) &= (s_1, \dots, s_{i-1}, 1, s_{i+1}, \dots, s_c, q) \\ &\quad \text{if } s_j > 0, j = 1, \dots, i-1, \text{ and } s_i = 0 \\ &= (s_1, \dots, s_c, q + 1) \\ &\quad \text{if } s_j > 0, j = 1, \dots, c, \text{ and } q < K - c \\ &= a \quad \text{if } q = K - c \\ t_A(a) &= a. \end{aligned}$$

Now consider the service-stage completion events. Whenever this event occurs the channels move to the next stage unless it is idle

$$\begin{aligned}
t_{C_i}(s_1, \dots, s_c, q) &= (s_1, \dots, s_i+1, \dots, s_c, q) \\
&\quad \text{if } 0 < s_i < k \\
&= (s_1, \dots, 1, \dots, s_c, q-1) \\
&\quad \text{if } s_i = k \text{ and } q > 0 \\
&= (s_1, \dots, 0, \dots, s_c, q) \\
&\quad \text{if } s_i = k \text{ and } q = 0, \text{ or if } s_i = 0 \\
t_{C_i}(a) &= a
\end{aligned}$$

This completes the SERT description of the model. It is a straightforward programming task to construct the rate and target vectors corresponding to the lexicographically ordered state space and to compute transient state probabilities using the randomization algorithm. There will be two sources of probability loss: the  $\epsilon$  probability lost due to truncation of the infinite series (4), and the probability absorbed in  $a$ ,  $\pi_a(t)$ . The total error on all probabilities computed will be bounded by  $-(\epsilon + \pi_a(t))$  and 0. If this bound is not satisfactory,  $\epsilon$  can be decreased and  $K$  increased.

### 5.3 A Reliability Example: Parallel System with Component Repair

Consider a system of  $n$  components in a parallel configuration. Initially all components are operating. The components have independent exponentially distributed lifetimes; the failure rate of the  $i$ th component is  $\lambda_i$ . Upon failure the component is repaired. The repair time has an independent generalized Erlang distribution with  $k_i$  stages and rate  $\mu_{i,j}$  of completion of the  $j$ th stage of repair,  $j = 1, 2, \dots, k_i$ . After repair

is completed the component functions for another independent exponential ( $\lambda_i$ ) period of time before failing again. System failure occurs when all  $n$  components are simultaneously under repair. Let  $L$  equal the time until system failure. We would like to compute  $\Pr\{L \leq t\}$ . Analytical and approximate aspects of this and related problems are given by Brown [1975] and Keilson [1975].

First let us consider the  $i$ th component. This component can be in any of  $k_i + 1$  states ( $s_i = 0, 1, 2, \dots, k_i$ ), where state 0 corresponds to a functioning unit and  $j$  corresponds to the  $j$ th stage of repair,  $j = 1, 2, \dots, k_i$ . The Markov process describing the  $i$ th component has generator

$$Q_i = \begin{pmatrix} -\lambda_i & \lambda_i & & & \\ & -\mu_{i,1} & \mu_{i,1} & & \\ & & -\mu_{i,2} & \mu_{i,2} & \\ & & & \ddots & \ddots \\ \mu_{i,k} & & & & -\mu_{i,k_i} \end{pmatrix}.$$

This process has one basic type of transition: "change." Thus we will be able to describe the evolution of the  $n$ -component system with  $n$  events:  $E = \{C_1, C_2, \dots, C_n\}$ , where  $C_i$  is the event of "change" of the  $i$ th component.

Now let us consider the state of the entire system. Let

$$s = (s_1, s_2, \dots, s_n)$$

where  $s_i$  equals the state of the  $i$ th component,  $s_i = 0, 1, 2, \dots, k_i$ , for  $i = 1, \dots, n$ . Let

$$S^* = \{(s_1, \dots, s_n) : 0 \leq s_i \leq k_i, i = 1, \dots, n\}$$

$$S_F = \{(s_1, \dots, s_n) : 1 \leq s_i \leq k_i, i = 1, \dots, n\}$$

and

$$S_0 = S^* - S_F .$$

The subset  $S_F$  is the set of all states corresponding to system failure and the subset  $S_0$  is the set of all states corresponding to an operating system. Assume that  $S_0$  is ordered lexicographically. Let  $\{f\}$  be a single absorbing state corresponding to system failure. The state space for this process can be described as

$$S = S_0 \cup \{f\} .$$

Now let us consider the collection  $R$  of rate vectors, one for each event type in  $E$ . Consider the event  $C_i$ . The rate vector (or function) for this event is

$$r_{C_i}(s_1, s_2, \dots, s_n) = \lambda_i \quad \text{if } s_i = 0$$

$$r_{C_i}(s_1, s_2, \dots, s_n) = \mu_{i, s_i} \quad \text{if } s_i > 0$$

$$r_{C_i}(f) = 0$$

Finally, the target vectors are

$$\begin{aligned} t_{C_i}(s_1, \dots, s_n) &= (s_1, \dots, s_i+1, \dots, s_n) \\ &\quad \text{if } 0 < s_i < k \text{ or} \\ &\quad \text{if } s_i < k \text{ and } s_j = 0 \text{ for some } i \neq j \\ &= (s_1, \dots, 0, \dots, s_n) \\ &\quad \text{if } s_i = k_i \\ &= f \\ &\quad \text{if } s_i = 0 \text{ and } s_j > 0 \text{ for all } j \neq i \end{aligned}$$

$$t_{C_i}(f) = f$$

This completes the SERT description of the process. It can be programmed and the randomization technique used to compute  $\pi_f(t)$ , which will equal  $\Pr\{L \leq t\}$ .

## 6. COMPUTATION OF OTHER TRANSIENT QUANTITIES

Using the randomization construction, useful computational formulae can be developed for other transient quantities: (i) expected time averages, (ii) first passage time distributions, and (iii) expected number of events of a certain type occurring during a time interval.

### 6.1 Expected Time Averages

Suppose  $\{X(t), t \geq 0\}$  is a continuous time Markov process with generator  $Q$ , where  $X(t)$ , the state that the system is in at time  $t$ , can take values  $s = 0, 1, 2, \dots, N$ . Further suppose that  $\Lambda = \max |q_s| < \infty$ , i.e., the process is uniformizable. Then from the preceding discussion, the process can be represented as a discrete time Markov chain on an underlying Poisson process, with rate  $\Lambda$  and transition matrix  $\tilde{P} = Q/\Lambda + I$ . Let  $\{N(t), t \geq 0\}$  be the counting process of the underlying Poisson process, i.e.,

$$N(t) = \# \text{ occurrences of Poisson process in } [0, t]$$

and let  $\{T_n, n = 1, 2, \dots\}$  be the times of occurrence of this Poisson process. (Then  $T_n = \min(t : N(t) \geq n)$ , etc.) The discrete time process  $\{Y_n, n = 0, 1, 2, \dots\}$ , then, is the state of the system just after the  $n$ th occurrence of the Poisson ( $\Lambda$ ) process. Thus  $\{Y_n, n = 0, 1, \dots\}$

is the discrete time Markov chain with transition matrix  $\tilde{P}$ . With this terminology we can now proceed to the quantities of interest.

$$\text{THEOREM 1. } E\left[\int_0^t X(\tau) d\tau\right] = \sum_{n=0}^{\infty} \left[ \left( \sum_{j=0}^n EY_j \right) / (n+1) \right] (e^{-\Lambda t} (\Lambda t)^n) / n! .$$

$$\begin{aligned} \text{Proof: } E\left[\int_0^t X(\tau) d\tau\right] &= \sum_{n=0}^{\infty} E\left[\int_0^t X(\tau) d\tau \mid N(t) = n\right] P(N(t) = n) \\ &= \sum_{n=0}^{\infty} E[Y_0 T_1 + Y_1 (T_2 - T_1) + \dots + Y_n (t - T_n) \mid N(t) = n] P(N(t) = n) \\ &= \sum_{n=0}^{\infty} E\left[\sum_{j=0}^n Y_j (T_{j+1} - T_j) \mid N(t) = n\right] P(N(t) = n) \\ &= \sum_{n=0}^{\infty} \sum_{j=0}^n E[Y_j (T_{j+1} - T_j) \mid N(t) = n] P(N(t) = n) \\ &= \sum_{n=0}^{\infty} \sum_{j=0}^n EY_j E[T_{j+1} - T_j \mid N(t) = n] P(N(t) = n) \\ &= \sum_{n=0}^{\infty} \sum_{j=0}^n EY_j 1/(n+1) P(N(t) = n) \\ &= \sum_{n=0}^{\infty} \left[ \left( \sum_{j=0}^n EY_j \right) / (n+1) \right] (e^{-\Lambda t} (\Lambda t)^n) / n! \end{aligned} \quad \text{Q.E.D.}$$

The associated computational formula in terms of the transient probability,  $\phi$ , for the Markov chain will be

$$\begin{aligned} E\left[\int_0^t X(\tau) d\tau\right] &= \sum_{n=0}^{\infty} \sum_{j=0}^n EY_j (e^{-\Lambda t} (\Lambda t)^n) / (n+1)! \\ &= \sum_{n=0}^{\infty} \left[ \sum_{j=0}^n \sum_{s \in S} s \phi_j(s) \right] (e^{-\Lambda t} (\Lambda t)^n) / (n+1)! . \end{aligned}$$

The above results can be generalized in the obvious straightforward way to computational formulae for

$$E[\int_0^t f(X(\tau))d\tau]$$

where  $f$  is any real valued function on the state space of the process.

As an example, consider a machine repair problem. Let  $X(\tau)$  equal the number of good machines available at time  $\tau$ . Let  $f(x) = \min(x, M)$ , where  $M$  is the desired number of operating machines. Then

$$E[\int_0^t \min(X(\tau), M)d\tau]$$

is the expected amount of machine operating time achieved during  $[0, t]$ .

As a similar example, consider a fleet of aircraft of which  $M$  will be kept in flight operations if  $M$  are available. Then the integral represents the amount of flight operation time achieved by the fleet during  $[0, t]$ . If  $T_s$  equals the sum of deterministic flight time per sortie and the deterministic inter-sortie ground time, then

$$E[\int_0^t \min(X(\tau), M)d\tau] / T_s$$

equals the expected number of sorties flown in  $[0, t]$ . (This assumes that the denominator is much less than the numerator, otherwise it is only an approximation unless  $t$  is an integer multiple of sortie time, in which case it is always exact.)

Other performance measures that can be computed in a similar way include the expected time the process spends in a particular subset of the state space, say  $A$ , during  $[0, t]$ :

$$E[\int_0^t 1_A(X(\tau))d\tau]$$

where  $1_A(\cdot)$  is the indicator function of the set  $A$ . If  $A$  indicates the set of states for which an availability requirement is satisfied, then the integral represents the amount of time during  $[0, t]$  that the requirement is met.

## 6.2 First Passage Time Distributions

Suppose  $\{X(t), t \geq 0\}$  is a Markov process on the finite state space  $S$ . Let  $A$  be a subset of  $S$  and  $T_A$  the first passage time,

$$T_A = \min\{t : X(t) \in A\}.$$

The randomization technique can be used to compute passage times as follows: define an associated process  $\{X_A(t), t \geq 0\}$  on the state space  $S - A \cup \{a\}$  with generator  $Q_A$  such that  $q_{A;i,j} = q_{i,j}$  for  $i, j \notin A$ ,  $q_{A;i,a} = \sum_{j \in A} q_{i,j}$  for  $i \notin A$ , and  $q_{A;a,j} = 0$  for all  $j$ . Thus the set  $A$  is collapsed down to one absorbing state  $a$ . The transition intensities are identical except for 0 rate out of  $A$ . Thus

$$P\{T_A \leq t\} = P\{X_A(t) = a\}$$

and the first passage probability (left-hand side of the equation) is equivalent to computing a transient state probability for a Markov process (generator  $Q_A$ ), which can be computed using the randomization algorithm. Melamed and Yadin [1980,1981] use this approach to compute sojourn times for queuing networks.

## 6.3 Expected Number of Events in an Interval

Another performance measure of interest (especially when a cost function is being considered) might be the expected number of a particular type or class of event which occur in a time interval  $[0, t]$ . For

example, in a reliability problem we might be interested in the expected number of failures occurring in  $[0, t]$ .

Let  $e_j \in E$  and suppose we are computing  $\phi(n)$ 's recursively, where  $\phi_s(n) = P(X(T_n) = s) = P(Y_n = s)$ . Then  $\phi_s(n) \cdot p_j(s) = P(Y_n = s \text{ and next event is } e_j)$ . Thus the inner product  $\sum_{s \in S} \phi_s(n) \cdot p_j(s)$  equals  $P(e_j \text{ occurs at } T_{n+1})$ , which also equals the expected number of  $e_j$ 's in  $(T_n, T_{n+1}]$ . Thus

$$\begin{aligned}
 & E[\text{number of } e_j \text{'s in } [0, t]] \\
 &= \sum_{n=0}^{\infty} E[\text{number of } e_j \text{'s in } [0, t] \mid N(t) = n] P[N(t) = n] \\
 &= \sum_{n=0}^{\infty} E[\text{number of } e_j \text{'s in } [0, T_n] \mid N(t) = n] P[N(t) = n] \\
 &= \sum_{n=0}^{\infty} \sum_{k=0}^{n-1} E[\text{number of } e_j \text{'s in } (T_k, T_{k+1}] \mid N(t) = n] P[N(t) = n] \\
 &= \sum_{n=0}^{\infty} \sum_{k=0}^{n-1} \sum_{s \in S} \phi_s(k) \cdot p_j(s) P[N(t) = n] .
 \end{aligned}$$

## 7. A GENERAL IMPLEMENTATION FOR PROCESSES WITH SPARSE GENERATORS

The SERT approach works well if each event can occur while the system is in an arbitrary state or if there are only a few states where a given event cannot occur. For example, in the machine repair system, a repair cannot occur when all machines are functional and a failure cannot occur when all machines are in repair; otherwise each event can occur in all remaining states, and SERT is quite efficient. However, there are sparse systems that are not amenable to this approach, e.g.,

$$Q = \begin{pmatrix} -1 & 1 & 0 & 0 & 0 \\ 0 & -6 & 3 & 3 & 0 \\ 4 & 4 & -20 & 6 & 6 \\ 0 & 5 & 7 & -12 & 0 \\ 0 & 0 & 0 & 2 & -2 \end{pmatrix}. \quad (11)$$

In this Markov process only 10 of the possible 20 transitions occur between the five states; however, the SERT description would require four events because state 3 has four possible transitions. Four events on a state space of five states will have  $4 \times 5 = 20$  transitions, and hence not exploit the sparseness at all.

There is a modification of the SERT approach that exploits the sparseness of  $Q$  to the fullest degree. It is based on a single target vector  $\underline{t}^*$  and a single rate vector  $\underline{r}^*$ : Suppose the state space is ordered into a one-dimensional set with  $N$  states,  $S = \{1, 2, \dots, N\}$ . For each state  $s$  in  $S$  let  $T_s = \{t_{s,1}, t_{s,2}, \dots, t_{s,n_s}\}$  be the set of states to which a transition is possible, i.e.,  $T_s = \{i \in S : q_{s,i} > 0, i \neq s\}$ , and  $n_s$  equals the number of such states. We can define a single target vector  $\underline{t}^*$  with  $N + \sum_{s=1}^N n_s$  components. Letting  $m_j = j + \sum_{i=1}^{j-1} n_i$ ,  $j = 1, \dots, N$ , define

$$\begin{aligned} t_{m_j}^* &= n_j \\ t_{m_j+k}^* &= t_{j,k} \\ r_{m_j}^* &= -q_j \\ r_{m_j+k}^* &= q_{j,t_{j,k}} \\ \tilde{p}_{m_j}^* &= 1 - q_j / \Lambda \\ \tilde{p}_{m_j+k}^* &= q_{j,t_{j,k}} / \Lambda \end{aligned}$$

$k = 1, 2, \dots, n_j$ ,  $j = 1, 2, \dots, N$ . All the information in a generator  $Q$  will be captured in the vectors  $\underline{r}^*$  and  $\underline{t}^*$ . For the generator in equation (11) this representation is given by

$$N = 5,$$

$$\Lambda = 20,$$

$$n_1 = 1, \quad n_2 = 2, \quad n_3 = 4, \quad n_4 = 2, \quad n_5 = 1, \quad N + \sum_{i=1}^N n_i = 15,$$

$$T_1 = \{2\}, \quad T_2 = \{3, 4\}, \quad T_3 = \{1, 2, 4, 5\}, \quad T_4 = \{2, 3\}, \quad T_5 = \{4\},$$

$$\underline{t}^* = (1, 2, 2, 3, 4, 4, 1, 2, 4, 5, 2, 2, 3, 1, 4),$$

$$\underline{r}^* = (-1, 1, -6, 3, 3, -20, 4, 4, 6, 6, -12, 5, 7, -2, 2),$$

$$\tilde{\underline{p}}^* = (.95, .05, .7, .15, .15, 0, .2, .2, .3, .3, .4, .25, .35, .9, .1).$$

This provides an economical way of storing a sparse generator  $Q$  of moderate or large size.

The transient distributions  $\phi(n)$  of the Markov chain on the underlying Poisson must be calculated. As before, this is done recursively using

$$\phi(n+1) = \phi(n)\tilde{P}. \quad (12)$$

The representation of  $Q$  with  $\underline{t}^*$  and  $\underline{r}^*$  yields a representation of  $\tilde{P}$  in terms of  $\underline{t}^*$  and  $\tilde{\underline{p}}^*$  that makes recursive calculation of  $\phi_n$  very easy.

Letting  $\phi_j(n+1)$  be PHINEW(J),  $\phi_j(n)$  be PHIOLD(J),  $t_i^*$  be TSTAR(I), and  $\tilde{p}_i^*$  be PSTAR(I), a FORTRAN routine for equation (12) is:

```

      DO 1 J = 1,N
1    PHINEW(J) = 0.0
      I = 0
      DO 3 J = 1,N
      I = I + 1
      PHIJ = PHIOLD(J)
      PHINEW(J) = PHINEW(J) + PHIJ * PSTAR(I)
      MJ = TSTAR(I)
      IF MJ.EQ.0 GO TO 3
      DO 2 K = 1,MJ
      I = I + 1
      TJK = TSTAR(I)
2    PHINEW(TJK) = PHINEW(TJK) + PHIJ * PSTAR(I)
3    CONTINUE

```

This representation and algorithm totally exploit any sparseness in  $Q$ .

For large systems the easier approach may often be a *SERT* description of the process. If a large degree of sparseness remains in the *SERT* description, i.e., many rates are 0, the set of target vectors  $T$  can be transformed into one vector  $\tilde{t}^*$  and the set of rate vectors  $R$  can be transformed into one vector  $\tilde{r}^*$  and the general approach used.

In some instances a hybrid approach might be best. If some events in the *SERT* description can occur at all states, these events can be modeled with individual rate and target vectors. If other events in the *SERT* description are impossible at many states, then these events should be pooled into one general "miscellaneous" event described by  $\tilde{t}^*$  and  $\tilde{r}^*$ .

In closing, we note that we are concerned with systems in which a large number of occurrences of the Poisson process will happen during the transient time interval. Thus our primary concern is the rapid calculation of equation (12). We are willing to spend a little extra time in setting up a one-dimensional state space and rate and target vectors in order to achieve maximum speed in this calculation.

## 8. CONCLUSIONS AND SUMMARY

We have presented a review of and some extensions to the randomization technique for computing transient quantities for Markov processes. We have introduced the general modeling approach of SERT which seems to have rather broad applicability and is a general and efficient means of implementing the randomization technique. Gross and Miller (1982) have used the SERT/Randomization approach on multi-echelon repairable item provisioning models. Miller (1982) has used SERT and the sparseness approach of Section 7 in reliability calculations for fault-tolerant reconfigurable computer systems.

The SERT approach has the advantage that it is general and applies to many systems. However, implementations of the randomization technique tailored to a specific system may be more efficient. In the latter two examples of Section 5 it is clear that various improvements could be made in the implementation by treating special cases. For example, in the  $M/E_k/c$  queue it is unnecessary to store entire rate vectors because they are all constant. Grassmann (1982) has written an efficient package to compute state probabilities for queuing networks. Melamed and Yadin

(1981) have developed an efficient algorithm for sojourn time distributions in queuing networks. In spite of this tradeoff between general and specific algorithms, it appears that SERT is a good starting point for the algorithmic analysis of transient Markov processes.

#### ACKNOWLEDGMENTS

The authors gratefully acknowledge the fine programming assistance of Mr. A. R. Balana and Mr. L. Kioussis. This research was supported in part by the Air Force Logistics Management Center under Contract SCEEE/AFLMC/80-6; Gross was also supported by the Office of Naval Research, Program in Logistics Contract N00014-75-C-0729, Project NR 347 020; Miller was also supported by the National Aeronautics and Space Administration under Grant NAG-1-179.

## REFERENCES

- ARSHAM, H., A. R. BALANA, and D. GROSS. 1981. Numerical Methods for Transient Solutions of Machine Repair Problems, Technical Paper Serial T-436, Institute for Management Science and Engineering, The George Washington University.
- BALANA, A. R., D. GROSS, and D. R. MILLER. 1982. Analytical Models for Transient Behavior of Repairable Item Provisioning, Technical Paper Serial T-462, Institute for Management Science and Engineering, The George Washington University.
- BROWN, M. 1975. The First Passage Time Distribution for a Parallel Exponential System with Repair. In *Reliability and Fault Tree Analysis* (R. E. Barlow, J. B. Fussell, and N. D. Singpurwalla, Eds.), 365-396, SIAM, Philadelphia.
- GINLAR, E. 1975. *Introduction to Stochastic Processes*. Prentice-Hall, Englewood Cliffs, New Jersey.
- COHEN, J. W. 1969. *The Single Server Queue*. North Holland, Amsterdam.
- FELLER, W. 1971. *An Introduction to Probability Theory and Its Applications, Vol. II*, 2 Ed. Wiley, New York.
- GRASSMANN, W. 1977. Transient Solutions in Markovian Queueing Systems. *Computers & Operations Res.* 4, 47-56.
- GRASSMANN, W. 1982. The QUE Package, University of Saskatchewan, Saskatoon, Canada.
- GROSS, D., and C. M. HARRIS. 1974. *Fundamentals of Queueing Theory*. Wiley, New York.

- GROSS, D., L. KIOUSSIS, and D. R. MILLER. 1982. Analytical Models for Transient Behavior of Multi-echelon Repairable Item Provisioning Systems with Finite Calling Populations and Finite Repair Capacities, Technical Paper Serial T-466, Institute for Management Science and Engineering, The George Washington University.
- GROSS, D., and D. R. MILLER. 1982. Multi-echelon Repairable Item Provisioning in a Time Varying Environment Using the Randomization Technique, Technical Paper Serial T-468, Institute for Management Science and Engineering, The George Washington University.
- HEYMAN, D. P., and M. J. SOBEL. 1982. *Stochastic Models in Operations Research, Vol. I*. McGraw-Hill, New York.
- HOWARD, R. 1960. *Dynamic Programming and Markov Processes*. Wiley, New York.
- JENSEN, A. 1953. Markoff Chains as an Aid in the Study of Markoff Processes. *Skandinavisk Aktuarietidskrift* 36, 87-91.
- KARLIN, S., and H. M. TAYLOR. 1975. *A First Course in Stochastic Processes*, 2 Ed. Academic Press, New York.
- KEILSON, J. 1975. Systems of Independent Markov Components and Their Transient Behavior. In *Reliability and Fault Tree Analysis* (R. E. Barlow, J. B. Fussell, and N. D. Singpurwalla, eds.), 351-364. SIAM, Philadelphia.
- KEILSON, J. 1979. *Markov Chain Models--Rarity and Exponentiality*. Springer-Verlag, New York.
- KEILSON, J., and A. KESTER. 1977. Monotone Matrices and Monotone Markov Processes. *Stochastic Processes and Their Applications* 5, 231-241.

- KIRSTEIN, B. M. 1977. Monotonicity and Comparability of Time-homogeneous Markov Processes with Discrete State Space. *Math. Operationsforsch. Statist.* 7, 151-168.
- LIU, M. L. 1966. A Novel Method of Evaluating Transient Response. *Proc. IEEE* 54, 20-23.
- LIPPMAN, S. 1975. Applying a New Device in the Optimization of Exponential Queueing Systems. *Opns. Res.* 23, 687-710.
- MARON, M. J. 1982. *Numerical Analysis--A Practical Approach*. Macmillan, New York.
- MELAMED, B., and M. YADIN. 1980. Randomization Procedures in the Computation of Cumulative Time Distributions Over Discrete State Markov Processes, preprint.
- MELAMED, B., and M. YADIN. 1981. Numerical Computation of Sojourn Time Distributions in Queueing Networks, preprint.
- MILLER, D. R. 1982. Reliability Calculation Using Randomization for Markovian Fault-tolerant Computing Systems, in preparation.
- MOLER, C., and C. VAN LOAN. 1978. Nineteen Dubious Ways to Compute the Exponential of a Matrix. *SIAM Review* 20, 801-836.
- ROSS, S. M. 1970. *Applied Probability Models with Optimization Applications*. Holden-Day, San Francisco.
- SERFOZO, F. R. 1979. An Equivalence Between Continuous and Discrete Time Markov Decision Processes. *Opns. Res.* 27, 616-620.
- SONDERMAN, D. 1980. Comparing Semi-Markov Processes. *Mathematics of Operations Research* 5, 110-119.
- VEINOTT, A. F. 1969. Discrete Dynamic Programming with Sensitive Discount Optimality Criteria. *Ann. Math. Statist.* 40, 1635-1660.

**THE GEORGE WASHINGTON UNIVERSITY**  
**Program in Logistics**  
**Distribution List for Technical Papers**

The George Washington University Office of Sponsored Research Gelman Library Vice President H. F. Bright Dean Harold Liebowitz Dean Henry Solomon	Armed Forces Industrial College	Case Western Reserve University Prof B. V. Dean Prof M. Mesarovic
ONR Chief of Naval Research (Codes 200, 434) Resident Representative	Armed Forces Staff College	Cornell University Prof R. E. Bechhofer Prof R. W. Conway Prof Andrew Schultz, Jr.
OPNAV OP-40 DCNO, Logistics Navy Dept Library NAVDATA Automation Cmd	Army War College Library Carlisle Barracks	Cowles Foundation for Research in Economics Prof Martin Shubik
Naval Aviation Integrated Log Support	Army Cmd & Gen Staff College	Florida State University Prof R. A. Bradley
NARDAC Tech Library	Army Logistics Mgt Center Fort Lee	Harvard University Prof W. G. Cochran Prof Arthur Schleifer, Jr.
Naval Electronics Lab Library	Commanding Officer, USALDSRA New Cumberland Army Depot	Princeton University Prof A. W. Tucker Prof J. W. Tukey Prof Geoffrey S. Watson
Naval Facilities Eng Cmd Tech Library	Army Inventory Res Ofc Philadelphia	Purdue University Prof S. S. Gupta Prof H. Rubin Prof Andrew Winston
Naval Ordnance Station Louisville, Ky. Indian Head, Md.	Army Trans Material Cmd TCMAC-ASDT	Stanford University Prof T. W. Anderson Prof Kenneth Arrow Prof G. B. Dantzig Prof F. S. Hillier Prof D. L. Iglehart Prof Samuel Karlin Prof C. J. Lieberman Prof Herbert Solomon Prof A. F. Veinott, Jr.
Naval Ordnance Sys Cmd Library	Air Force Headquarters AFADS-3 LEXY SAF/ALG	University of California, Berkeley Prof R. E. Barlow Prof D. Gale Prof Jack Kiefer
Naval Research Branch Office Boston Chicago New York Pasadena San Francisco	Griffiss Air Force Base Reliability Analysis Center	University of California, Los Angeles Prof R. R. O'Neill
Naval Ship Eng Center Philadelphia, Pa.	Gunter Air Force Base AFLMC/XR	University of North Carolina Prof W. L. Smith Prof M. R. Leadbetter
Naval Ship Res & Dev Center	Maxwell Air Force Base Library	University of Pennsylvania Prof Russell Ackoff
Naval Sea Systems Command PMS 30611 Tech Library Code 073	Wright-Patterson Air Force Base AFLC/OA Research Sch Log AFALD/XR	University of Texas Institute for Computing Science and Computer Applications
Naval Supply Systems Command Library Operations and Inventory Analysis	Defense Technical Info Center	Yale University Prof F. J. Anscombe Prof H. Scarf
Naval War College Library Newport	National Academy of Sciences Maritime Transportation Res Bd Lib	Prof Z. W. Birnbaum University of Washington
BUPERS Tech Library	National Bureau of Standards Dr B. H. Colvin Dr Joan Rosenblatt	Prof B. H. Bissinger The Pennsylvania State University
FMSO	National Science Foundation	Prof Seth Boder University of Michigan
USN Ammo Depot Earle	National Security Agency	Prof G. E. Box University of Wisconsin
USN Postgrad School Monterey Library Dr Jack R. Borsting Prof C. R. Jones	Weapons Systems Evaluation Group	Dr Jerome Bracken Institute for Defense Analyses
US Coast Guard Academy Capt Jimmie D. Woods	British Navy Staff	
US Marine Corps Commandant Deputy Chief of Staff, R&D	National Defense Hdqtrs, Ottawa Logistics, OR Analysis Estab	
Marine Corps School Quantico Landing Force Dev Ctr Logistics Officer	American Power Jet Co George Chernowitz	
	General Dynamics, Pomona	
	General Research Corp Library	
	Logistics Management Institute Dr Murray A. Geisler	
	Rand Corporation Library Mr William P. Hutzler	
	Carnegie-Mellon University Dean H. A. Simon Prof G. Thompson	

Continued

Prof A. Charnes  
University of Texas

Prof H. Chernoff  
Mass Institute of Technology

Prof Arthur Cohen  
Rutgers - The State University

Mr Wallace M. Cohen  
US General Accounting Office

Prof C. Derman  
Columbia University

Prof Masao Fukushima  
Kyoto University

Prof Saul I. Gass  
University of Maryland

Dr Donald P. Gaver  
Carmel, California

Prof Amrit L. Goel  
Syracuse University

Prof J. F. Hannan  
Michigan State University

Prof H. O. Hartley  
Texas A & M Foundation

Prof W. M. Hirsch  
Courant Institute

Dr Alan J. Hoffman  
IBM, Yorktown Heights

Prof John R. Isbell  
SUNY, Amherst

Dr J. L. Jain  
University of Delhi

Prof J. H. K. Kao  
Polytech Institute of New York

Prof W. Kruskal  
University of Chicago

Mr S. Kumar  
University of Madras

Prof C. E. Lemke  
Rensselaer Polytech Institute

Prof Loynes  
University of Sheffield, England

Prof Tom Maul  
Kowloon, Hong Kong

Prof Steven Nahmias  
University of Santa Clara

Prof D. B. Owen  
Southern Methodist University

Prof P. R. Parathasarathy  
Indian Institute of Technology

Prof E. Parzen  
Texas A & M University

Prof H. O. Posten  
University of Connecticut

Prof R. Remage, Jr.  
University of Delaware

Prof Hans Riedwyl  
University of Berne

Mr David Rosenblatt  
Washington, D. C.

Prof M. Rosenblatt  
University of California, San Diego

Prof Alan J. Rowe  
University of Southern California

Prof A. H. Rubenstein  
Northwestern University

Prof Thomas L. Saaty  
University of Pittsburgh

Dr M. E. Salvesson  
West Los Angeles

Prof Gary Scudder  
University of Minnesota

Prof Edward A. Silver  
University of Waterloo, Canada

Prof Rosedith Sitgreaves  
Washington, DC

LTC G. L. Slyman, MSC  
Department of the Army

Prof M. J. Sobel  
Georgia Inst of Technology

Prof R. M. Thrall  
Rice University

Dr S. Vajda  
University of Sussex, England

Prof T. M. Whitin  
Wesleyan University

Prof Jacob Wolfowitz  
University of South Florida

Prof Max A. Woodbury  
Duke University

Prof S. Zacks  
SUNY, Binghamton

Dr Israel Zang  
Tel-Aviv University